

Decomposing Petri nets

Julian Rathke, Paweł Sobociński, and Owen Stephens

ECS, University of Southampton, UK

Abstract. In recent work, the second and third authors introduced a technique for reachability checking in 1-bounded Petri nets, based on *wiring decompositions*, which are expressions in a fragment of the compositional algebra of *nets with boundaries*. Here we extend the technique to the full algebra and introduce the related structural property of *decomposition width* on directed hypergraphs. Small decomposition width is necessary for the applicability of the reachability checking algorithm. We give examples of families of nets with constant decomposition width and develop the underlying theory of decompositions.

Introduction

Model checking asynchronous systems is notoriously susceptible to state explosion. Historically, Petri nets are one of the most popular formalisms for modelling asynchronous systems. Several model checking problems reduce to checking reachability in (bounded) Petri nets, where state explosion manifests itself in the fact that the set of markings is exponential in the number of places. Our approach to the problem of state explosion is to check reachability of a net in a divide-and-conquer, dynamic programming style [11] by considering decompositions of the net into smaller subnets and checking reachability locally. Clearly, this approach relies heavily on a principled notion of Petri net decomposition, which is the topic of this paper.

In [9] the second author introduced a compositional algebra of 1-bounded Petri nets, called *nets with boundaries*, which was later extended by Bruni, Melgratti and Montanari [1] to cover P/T nets; see [2] for a complete exposition. A net with boundaries induces a labelled transition system (LTS) where the states correspond to the markings of the net and the transitions witness the firings of independent sets of net transitions. Following the process calculus tradition, the labels of LTS transitions describe synchronisations with the environment.

In recent work [11], the second and third author used this algebra to check reachability for 1-bounded nets. A decomposition of a net into an expression in the algebra of nets with boundaries is called a *wiring decomposition*—concretely, it is a tree, with internal nodes labelled by the two operations ‘;’ and ‘ \otimes ’ for composing nets with boundaries, and the leaves labelled with individual nets with boundaries. For the purposes of reachability, given a wiring decomposition, each component net’s LTS is considered as a non-deterministic finite automaton (NFA) with initial state the initial (local) marking and final state the desired (local) marking. Because the algebra is compositional, the NFA for the entire

net can be obtained by composing the NFAs of the individual component nets, following the structure of the wiring decomposition. This underlying algebra of NFAs (transition systems) is that of $\text{Span}(\text{Graph})$ [6].

If, given a net, a “good” wiring decomposition can be found then characterising communication between components will require small (w.r.t. the global statespace) amounts of information. Once reachability is checked locally, local statespace can be discarded and thus state-explosion circumvented. Exposing the regular structure of a net, moreover, allows repeated work to be avoided: memoisation of local reachability checks on small component nets leads to better performance. As a result, in some examples (see [11]) reachability checking is *linear* in the size of the net, even when *the length of the minimal firing sequence required to reach the desired marking is non-linear*. The approach can thus sometimes outperform classical techniques for checking reachability, for instance, those based on the unfolding technique, originally pioneered by McMillan [7].

The applicability of the technique described in [11] is thus closely related to the problem of obtaining wiring decompositions of nets. When translating a net with boundaries to an LTS, its size depends on two factors: (i) the number of places and (ii) the size of its boundaries. The size of the LTS statespace is typically exponential in the number of places, as states correspond to markings. The size of the set of LTS labels is exponential in the size of the boundary.

What is a “good” wiring decomposition? Recall that a wiring decomposition is a tree. Firstly, the leaves of this tree are subnets and, in order to keep the size of the LTSs manageable, each leaf should have few places, and a small boundary. Secondly, each subtree of the wiring decomposition should result in a net with a small boundary, to keep the size of the label set small when checking the compositions of subnets. Thirdly, the minimised statespaces of (NFAs of) subtrees should “grow slowly” towards the root, so that state explosion is avoided.

The first two conditions amount to a *structural property*¹ on the underlying net, considered as a directed hypergraph. We call this property *decomposition width*: a net (or directed hypergraph) has *decomposition width* k iff it has a wiring decomposition of width k . The third condition is a *semantic property*: in particular, a net can have several decompositions of equal width that perform differently with respect to the third criterion. Several examples are given in [11].

In this paper, we concentrate on the structural property of decomposition width. We make use of the full algebra of nets with boundaries [2], which allows us to cover more examples than in [11] where we considered a restricted variant. We discover that sparsely connected nets, “tree-like” nets, but also cliques and related “densely” connected nets are all examples of families of nets that admit decompositions of small width. By this we mean that there is some k such that the entire family of nets (of arbitrary size) has decomposition width k . We also give an example of a family of grid nets that we conjecture not to admit bounded decomposition. Decomposition width is thus different to parameters which have

¹ Analogously to how pathwidth and treewidth are structural properties of undirected graphs. Treewidth is well known in the CONCUR community through Courcelle’s theorem [3].

previously been considered on nets, such as treewidth of the flow graph [8]; (like treewidth, grids seem problematic, but unlike treewidth, cliques are not.)

Concretely, the contributions of this paper are:

- The full algebra of nets with boundaries [9,11] is used with the reachability technique of [11]. We thus extend the applicability of the technique to examples such as clique nets.
- The structural property of decomposition width on nets (or, more generally, on directed hypergraphs) is introduced.
- The theory of wiring decompositions is developed, which allows us to give lower bounds on boundary sizes in certain decompositions.

Structure of the paper. In §2 we recall and generalise the definition of nets with boundaries. In §3 we introduce the notion of *decomposition width*, and explain its central role in the performance of our technique, which we briefly recap in §3.1. We discuss an extension to the previously considered net algebra in §4, using the full algebra of nets with boundaries in order to apply our technique to more cases. In §5 we introduce the principles of decomposition, and use them to show lower bounds for the size of decompositions in certain nets.

1 Preliminaries

For $n \in \mathbb{N}$, let $[n] = \{0, 1, \dots, n-1\}$. Write 2^X for the powerset of X and $X + Y$ for the set $\{(x, 0) \mid x \in X\} \cup \{(y, 1) \mid y \in Y\}$.

Definition 1 (1-bounded Petri net). A net N is $(P, T, {}^\circ -, -^\circ)$ where

- P is the set of places, T is the set of transitions
- ${}^\circ -, -^\circ : T \rightarrow 2^P$ give, respectively, the pre- and post-sets of each transition.

We write $\text{places}(N)$ and $\text{trans}(N)$ for the place and transition sets, respectively, of N . Our underlying semantics is a step firing semantics where independent sets of transitions can be fired together; to minimise redundancy, we give the definition in (1) in the more general setting of nets with boundaries.

2 Nets with boundaries

A net with boundaries [9] is a Petri net together with two ordered sets of *boundary ports*, to which net transitions can connect. Nets with boundaries inherit the algebra of monoidal categories for composition. In this paper we expand upon the previous exposition of nets with boundaries in [9,11], by lifting the restriction of [11] that at most one transition can connect to any one place on a boundary.

Definition 2 (Net with boundaries). A net with boundaries $N : k \rightarrow l$ is $(P, T, k, l, {}^\circ -, -^\circ, \bullet -, -\bullet, \bowtie)$ where:

- $(P, T, {}^\circ -, -^\circ)$ is a 1-bounded Petri net

- $k, l \in \mathbb{N}$ are, respectively, the left and the right boundaries
- $\bullet - : T \rightarrow 2^{[k]}$ and $- \bullet : T \rightarrow 2^{[l]}$ connect each transition to, respectively, the left and the right boundary
- \bowtie is a contention relation (see Definition 4 below).

Isomorphism, $(N : k \rightarrow l) \cong (M : k \rightarrow l)$, is defined in the obvious way as bijections between place sets and transition sets that respect pre and post sets, boundary connections and contention. 1-bounded Petri nets N can be considered as nets with boundaries $N : 0 \rightarrow 0$ (with the minimal contention relation).

Remark 3. In [11] we assumed that for any $t \neq t' \in T$, $\bullet t \cap \bullet t' = \emptyset$ and $t^\bullet \cap t'^\bullet = \emptyset$; i.e. no two transitions connect to the same boundary port. In Sec. 4, we show that certain nets admit better decompositions without this restriction.

In order to leave out the assumption, we must recall the notion of *contention* between transitions, first proposed in [2]. Transitions in contention cannot fire concurrently. In ordinary nets, two transitions are in contention precisely when they compete for a resource, for instance they consume or produce a token at the same place. In nets with boundaries, connecting two transitions to the same boundary port is another source of contention. Examples and the mathematical foundations of contention are given in [10]. Roughly speaking, contention is “remembered” in compositions; this is needed in order to ensure that net composition is compatible with the composition of underlying transition systems.

Definition 4 (Contention Relation). For a net N , a reflexive, symmetric relation, \bowtie , on $\text{trans}(N)$ is said to be a contention relation, if for all $(t, u) \in \text{trans}(N) \times \text{trans}(N)$ where at least one of the following holds

$$(i) \circ t \cap \circ u \neq \emptyset \quad (ii) t^\circ \cap u^\circ \neq \emptyset \quad (iii) \bullet t \cap \bullet u \neq \emptyset \quad (iv) t^\bullet \cap u^\bullet \neq \emptyset.$$

then $t \bowtie u$.

Remark 5 (Graphical representation). See Fig. 2 and Fig. 5 for several simple examples of nets with boundaries. The graphical representation we use is non-standard and deserves an explanation: Concretely, each place is drawn as “directed,” having an *in* and *out* port. Transitions are undirected links that connect an arbitrary set of boundaries and place ports. The benefit of doing this is that links, which are connected together during composition, do not need to be directionally compatible in order to compose two nets. Instead, the places contain the firing direction information, localising the firing semantics to subcomponents. The preset of a transition is simply the set of places to which the transition is connected via the *out* port (a triangle pointing out of a place), symmetrically, its postset is the set of places to which the transition is connected via the *in* port (a triangle pointing into a place.) In order to distinguish individual transitions and increase legibility, transitions are drawn with a small perpendicular mark.

A transition set U is *mutually independent* (MI) if $\forall u, v \in U. u \bowtie v \Rightarrow u = v$. Contention can be lifted to *sets* of mutually independent transitions: $U \bowtie V$ iff

$\exists u \in U, v \in V. u \bowtie v$. Mutually independent transitions can fire concurrently: each net with boundaries $N : k \rightarrow l$ determines an LTS², $\llbracket N \rrbracket$, whose transitions witness the step firing semantics of the underlying net. The labels are pairs of binary strings of length k and l , respectively. The states are markings of N , denoted by $[N]_X$, where $X \subseteq \text{places}(N)$. The transition relation is defined³:

$$[N]_X \xrightarrow{\alpha/\beta} [N]_{X'} \Leftrightarrow \exists \text{ MI } U \subseteq T, {}^\circ U \subseteq X, U^\circ \cap X = \emptyset, \\ X' = (X \setminus {}^\circ U) \cup U^\circ, \bullet U = \alpha, U^\bullet = \beta. \quad (1)$$

In order to compose nets with boundaries along a common boundary, we recall the notion of *synchronisation*. For sets of transitions $U \subseteq T$ we abuse notation and write ${}^\circ U = \bigcup_{u \in U} {}^\circ u$, and similarly for U° , $\bullet U$ and U^\bullet .

Definition 6 (Synchronisations). A synchronisation between two nets with boundaries $M : l \rightarrow m$, $N : m \rightarrow n$ is a pair (U, V) , $U \subseteq \text{trans}(M)$ and $V \subseteq \text{trans}(N)$, of mutually independent sets of transitions, such that $U^\bullet = \bullet V$.

Synchronisations inherit an ordering from the subset ordering, pointwise: $(U, V) \subseteq (U', V') \stackrel{\text{def}}{=} U \subseteq U' \wedge V \subseteq V'$. The trivial synchronisation is (\emptyset, \emptyset) . A synchronisation (U, V) is minimal when it is not trivial, and for all $(U', V') \subseteq (U, V)$, then (U', V') is trivial or equal to (U, V) . Contention can be lifted to minimal synchronisations: $(U, V) \bowtie (U', V') \stackrel{\text{def}}{=} U \bowtie U' \vee V \bowtie V'$.

Given $M : l \rightarrow m$, $N : m \rightarrow n$, let $\text{Synch}(M, N)$ be the set of minimal synchronisations. We can now define the two ways of composing nets with boundaries.

Definition 7 (Composition along common boundary). The composition of nets $M : l \rightarrow m$ and $N : m \rightarrow n$, $M; N : l \rightarrow n$ has the following components:

- the set of places is $\text{places}(M) + \text{places}(N)$.
- the set of transitions is $\text{Synch}(M, N)$, the set of minimal synchronisations.
- $\forall (U, V) \in \text{Synch}(M, N), {}^\circ(U, V) \stackrel{\text{def}}{=} {}^\circ U + {}^\circ V$ and $(U, V)^\circ \stackrel{\text{def}}{=} U^\circ + V^\circ$.
- $\forall (U, V) \in \text{Synch}(M, N), \bullet(U, V) \stackrel{\text{def}}{=} \bullet U$ and $(U, V)^\bullet \stackrel{\text{def}}{=} V^\bullet$.
- Contention on minimal synchronisations as described in Definition 6.

Definition 8 (Tensor product). The tensor product of nets $M : l \rightarrow m$ and $N : k \rightarrow n$, $M \otimes N : l + k \rightarrow m + n$ has the following components:

- the set of places is $\text{places}(M) + \text{places}(N)$.
- the set of transitions is $\text{trans}(M) + \text{trans}(N)$.
- the preset, postset, and boundary maps are defined in the obvious way.
- transitions in $\text{trans}(M) + \text{trans}(N)$ are in contention exactly when they are in contention in either M or N .

² Originally described in Katis et al [5].

³ We equate binary strings of length k with subsets of $[k]$, in the obvious way.

Both ‘;’-composition and ‘ \otimes ’-composition are associative up-to isomorphism. In examples we will make use of an exponentiation notation: given $N : l \rightarrow l$, we write N^k for the ‘;’-composition of N with itself k -times: $N ; N ; \dots ; N$.

There are several compositionality results reported in [9,2,11] (e.g. Theorem 3.8 of [9]); essentially the idea is that firings of a composed net (as LTS transitions $\xrightarrow{\alpha/\beta}$) are in direct correspondence with firings ($\xrightarrow{\alpha/\gamma}$ and $\xrightarrow{\gamma/\beta}$) of components.

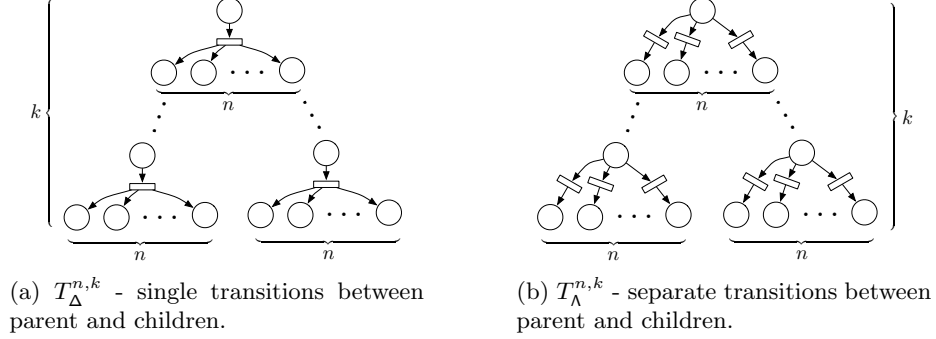


Fig. 1: Complete tree nets of depth k and width n .

Example 9. As an example of the use of the algebra of nets with boundaries, consider the net $T_{\Delta}^{n,k}$, where $k, n \geq 1$, in Fig. 1a. We can give a simple decomposition that relies on the components nets illustrated in Fig. 2. First, we define

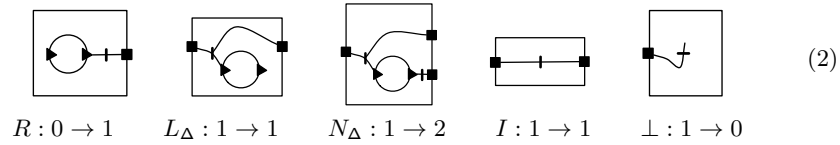


Fig. 2: Components used in the decomposition of $T_{\Delta}^{n,k}$.

the net with boundaries $B_{\Delta}^{n,k} : 1 \rightarrow 0$ by recursion on k :

$$B_{\Delta}^{n,k} \stackrel{\text{def}}{=} \begin{cases} L_{\Delta}^n ; \perp & \text{if } k = 1 \\ (N_{\Delta} ; (I \otimes B_{\Delta}^{n,i}))^n ; \perp & \text{if } k = i + 1 \end{cases} \quad (3)$$

whence it follows that $T_{\Delta}^{n,k} \cong R ; B_{\Delta}^{n,k}$. (4)

The decomposition of $T_{\Delta}^{2,2}$, following the definition in (4), is illustrated in Fig. 3; components enclosed with - - - are composed with ‘;’, while components enclosed with - - - are composed with ‘ \otimes ’.

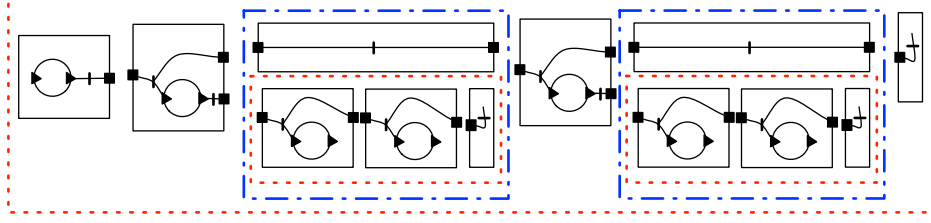


Fig. 3: Decomposition of $T_{\Delta}(2, 2)$.

3 Wiring Decompositions

To formalise the decomposition of nets with boundaries, such as that presented in Example 9, we introduce the concept of a *wiring decomposition*. A *wiring expression* is a syntactic term formed from the following grammar:

$$T ::= x \mid T ; T \mid T \otimes T$$

that is, a binary tree, with internal ‘;’ and ‘ \otimes ’ nodes and variables at the leaves.

A *variable assignment* \mathcal{V} is a map that takes variables to nets with boundaries. Given a pair (t, \mathcal{V}) of a wiring expression t and variable assignment \mathcal{V} , its semantics $\llbracket t \rrbracket_{\mathcal{V}}$ is a net with boundaries, defined inductively:

$$\llbracket x \rrbracket_{\mathcal{V}} \stackrel{\text{def}}{=} \mathcal{V}(x) \quad \llbracket t_1 ; t_2 \rrbracket_{\mathcal{V}} \stackrel{\text{def}}{=} \llbracket t_1 \rrbracket_{\mathcal{V}} ; \llbracket t_2 \rrbracket_{\mathcal{V}} \quad \llbracket t_1 \otimes t_2 \rrbracket_{\mathcal{V}} \stackrel{\text{def}}{=} \llbracket t_1 \rrbracket_{\mathcal{V}} \otimes \llbracket t_2 \rrbracket_{\mathcal{V}}$$

We implicitly assume that variable assignments are compatible with t : in the sense that only nets with a common boundary are composed; we omit the details, which are straightforward.

Definition 10. *Given a net $N : k \rightarrow l$, we say that the pair (t, \mathcal{V}) is a wiring decomposition of N if $\llbracket t \rrbracket_{\mathcal{V}} \cong N$.*

Example 11. A wiring decomposition of $T_{\Delta}^{n,k}$ can be obtained from (3) and (4) above by rewriting the equations as syntactic terms, with variables in place of each of the small component nets, and choosing a particular association for the ‘;’ and ‘ \otimes ’ expressions. We will see below that this particular choice of associativity is unimportant in terms of decomposition size but nonetheless has ramifications for the efficiency of our reachability checking algorithm (see [11] for examples).

3.1 Reachability via Compositionality

In this section we give a summary of the approach introduced in [11], where, given a 1-bounded Petri net, we decompose it using algebra of nets with boundaries to calculate reachability in divide-and-conquer style. However, the technique is only viable for nets for which we can find “small” decompositions.

As discussed in Sec. 2, each net with boundaries determines an LTS, witnessing its step semantics. For a given reachability problem, we can transform the LTS into a NFA, by letting the initial and final states of the NFA be those corresponding to the initial and final markings. Reachability then coincides with non-emptiness of the NFA’s language. To achieve a bounded statespace using our technique, we require that the considered nets admit “small” decompositions (the precise definition of which is presented Sec. 3.2.)

We rely on the compositionality of nets with boundaries in order to perform local checking of global reachability, w.r.t. interactions on a components’ boundaries: the NFA of a component net encodes the required “protocol” that the net must engage in with its environment in order to reach a (locally) final marking. Thus, to generate $NFA(x ; y)$, it suffices to generate $NFA(x)$, and $NFA(y)$ and compose them using a variant of the product construction: $(a, b) \xrightarrow{\alpha/\beta} (a', b')$ iff $\exists \gamma. a \xrightarrow{\alpha/\gamma} a' \wedge b \xrightarrow{\gamma/\beta} b'$.

Hiding internal computations improves the performance of our technique; we perform ϵ -closure on the obtained NFAs, identifying internal states that are distinguished only by transitions that do not alter the net’s protocol. Further, we avoid state explosion by minimising the NFA’s representation size, applying determinisation followed by DFA-minimisation to generate an automaton that recognises the same language, but with potentially simpler structure. Observe that after performing ϵ -closure and minimisation on the NFA of a net $N : 0 \rightarrow 0$ we have either the trivially accepting, or trivially rejecting automaton.

Furthermore, many nets have a repeated internal structure—several examples being presented in [11], and this paper. By exposing this repeated structure through decomposition, we avoid duplicating work, by employing memoisation such that conversion to NFA, or NFA composition is only performed once.

Example 12. Consider a decomposition of $T_{\Delta}^{2,2}$, as defined in (3) and (4), and illustrated in Fig. 3. Let the initial marking be a single token at the root place, and the final marking having only leaves marked. The minimal DFAs obtained from this decomposition are presented in Fig. 4⁵. For example, observe that $B_{\Delta}^{2,1}$ reaches its local accept state upon interacting once on its left boundary. Reachability is confirmed: the minimal DFA representing $T_{\Delta}^{2,2}$ is the trivial accepting automaton.

3.2 Decomposition width

As explained in the preceding section, the “size” of a decomposition is important for performance. We formalise this below.

⁴ Similarly, we can perform \otimes -composition on NFAs with a different modification of the standard product construction.

⁵ We have omitted error states if present. Labels indicate interaction with the boundaries: 00/1 is action on the right boundary, with no action on either left boundary. ‘*’ means either 0 or 1.

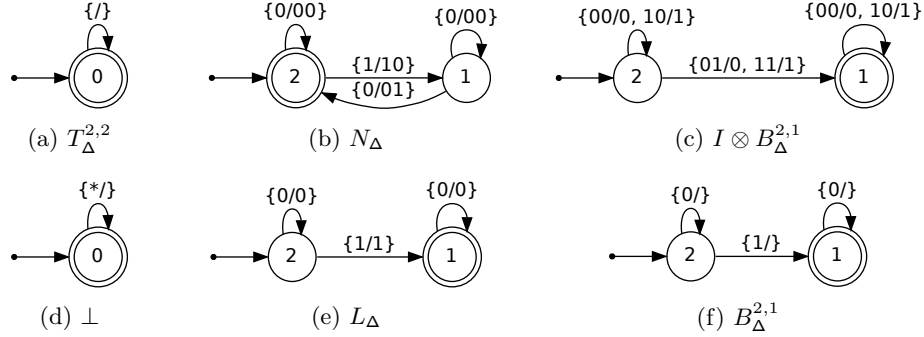


Fig. 4: Component NFAs of the right-associative $T_{\Delta}^{2,2}$ decomposition.

Definition 13 (Decomposition width). We say that a wiring decomposition, (t, \mathcal{V}) , of a net with boundaries has width $k \in \mathbb{N}$, if:

- (i) $\forall x \in t$, $\llbracket x \rrbracket_{\mathcal{V}} : l \rightarrow r$, with places P , satisfies $\max(l, |P|, r) \leq k$, and
- (ii) for all subexpressions t' of t , if $\llbracket t' \rrbracket_{\mathcal{V}} : l \rightarrow r$ then $\max(l, r) \leq k$.

A net has decomposition width k if it has a wiring decomposition of width k . A family of nets $\{N_i\}_{i \in I}$ has bounded decomposition width if there exists $k \in \mathbb{N}$ such that for all $i \in I$, N_i has decomposition width k .

Lemma 14 (Invariance w.r.t. associativity). Given a wiring decomposition, (t, \mathcal{V}) , of a net $N : l \rightarrow r$ that has width k , and given a wiring expression t' such that t' is equivalent to t up to associativity of ‘;’ and ‘ \otimes ’ then (t', \mathcal{V}) also has width k and $\llbracket t' \rrbracket_{\mathcal{V}} : l \rightarrow r$.

Proof. Write $t \sim t'$ for equivalence up to associativity and proceed by induction on the structure of t' . If t' is a variable then it is equal to t and hence the result follows.

Suppose that t' is an n -fold ‘ \otimes ’-composition of some t'_i for $1 \leq i \leq n$ such that t is also an n -fold ‘ \otimes ’-composition of some t_i with any other possible association with $t'_i \sim t_i$. By the induction hypothesis we see that each (t'_i, \mathcal{V}) has width k and $\llbracket t'_i \rrbracket_{\mathcal{V}} : l_i \rightarrow r_i$ where $l = (\sum_{1 \leq i \leq n} l_i) \leq k$ and $r = (\sum_{1 \leq i \leq n} r_i) \leq k$. Any subexpression of t' is either a subexpression of one of the t'_i (and hence satisfies boundedness) or some expression t'' containing a ‘ \otimes ’-composition of a subsequence I of the t'_i . The boundaries of (t'', \mathcal{V}) have size $l_I = \sum_I l_i \leq k$ and $r_I = \sum_I r_i \leq k$. Hence (t', \mathcal{V}) also has width k and $\llbracket t' \rrbracket_{\mathcal{V}} : l \rightarrow r$ as required. \square

Note that the algebra of nets with boundaries is actually an algebra of directed hypergraphs (that happens to be compositional w.r.t. the net semantics). Thus, the notion of decomposition width, introduced above, is—more generally—a structural property of directed hypergraphs.

Example 15. Consider the net $T_{\Delta}^{n,k}$ from Fig. 1a, decomposed in (3) and (4). For any n, k , this wiring decomposition has width 2: observe that every component net of $B_{\Delta}^{n,k}$ has at most one place and two boundary ports on either side

Furthermore, it is easy to confirm that at each internal node of the tree, two subtree nets are composed such that the resulting net has boundaries ≤ 2 , i.e. subexpressions have boundaries ≤ 2 . That is, a decomposition width of 2.

4 Harnessing the full algebra

In this section we use the full algebra of nets with boundaries in order to obtain decompositions of bounded width nets that do not have satisfactory decompositions using merely the subalgebra used in [11], described in Remark 3. Since, as explained in Sec. 3.1, a bounded decomposition width is a necessary condition for the applicability of our reachability checking approach, by doing so, we are able to extend its applicability to several natural families of nets.

Example 16. Consider the family of nets $T_\Lambda^{n,k}$ in Fig. 1b. These nets are similar to those discussed in Example 9, but with n *distinct* transitions from any non-leaf node to its children.

There is no way of obtaining a decomposition of bounded width with the restriction of Remark 3, i.e. at most one transition connected to each boundary port. To see why, assume we have a decomposition and consider the component that contains the root node: as we increase n one would have to either increase the size of the boundary or increase the number of places within the component. Without the restriction we can connect multiple transitions to the same boundary port, and so modify the construction of Example 9 to obtain a decomposition for $T_\Lambda^{n,k}$: Again, first define the component net $B_\Lambda^{n,k} : 1 \rightarrow 0$ by recursion on k :

$$B_\Lambda^{n,k} \stackrel{\text{def}}{=} \begin{cases} L_\Lambda^n ; \perp & \text{if } k = 1 \\ (N_\Lambda ; (I \otimes B_\Lambda^{n,i}))^n ; \perp & \text{if } k = i + 1 \end{cases} \quad (5)$$

whence we have that: $T_\Lambda^{n,k} \cong R ; B_\Lambda^{n,k}$. (6)

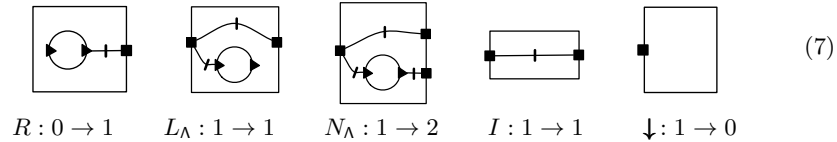


Fig. 5: Components used in the decomposition of $T_\Lambda^{n,k}$.

In addition to the decompositions in Examples 11 and 16 we will consider two other families of nets that are “densely” connected and show that they nevertheless have bounded decomposition width.

Example 17. Consider the clique net C_n : it has n places and $n \times (n - 1)$ transitions, one from each place to every other. An illustration of C_4 is given in Fig. 6a. It is easy to see that the flowgraph of C_n has treewidth $n - 1$, on the other hand C_n has decomposition width 2 for any n .

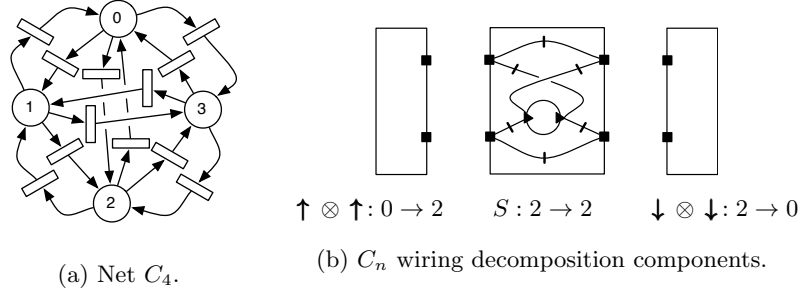


Fig. 6: Decomposing cliques.

The decomposition is simple and uses the components illustrated in Fig. 6b. Indeed, it is not difficult to see that $C_n \cong (\uparrow \otimes \uparrow) ; S^n ; (\downarrow \otimes \downarrow)$.

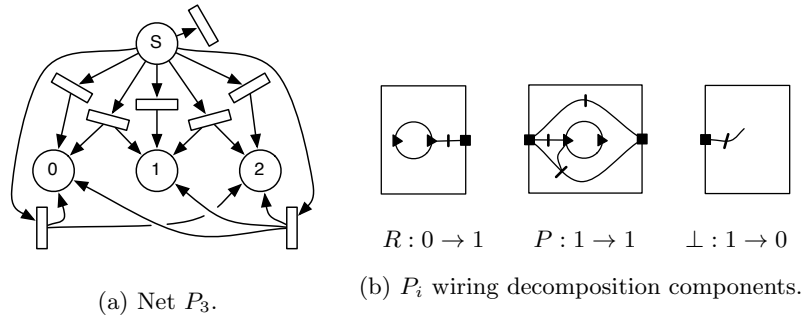


Fig. 7: Decomposing subset nets.

Example 18. Consider the net P_n , $n \geq 0$, with $n + 1$ places. There is a chosen place S , with the remaining places $0, 1, \dots, n - 1$, the elements of $[n]$. There are 2^n transitions in P_n , all with the single source S and targets the elements of $2^{[n]}$. See Fig. 7a for an illustration of P_3 . For any $n > 1$, P_n has a wiring decomposition of width 1: indeed, consider the components in Fig. 7b, then an easy calculation confirms that $P_n \cong R ; P^n ; \perp$.

Having extended the scope of our reachability technique to that of the full algebra of nets with boundaries, we are able to handle more examples, such as those presented in this section.

5 Principles of decomposition

In Examples 11, 16, 17 and 18 we exhibited several families of nets with bounded decomposition width. In this section we develop the theory of decompositions that will allow us to place lower bounds on the size of shared boundaries in certain decompositions. Taking these initial observations into consideration, we conjecture that the family of grid nets $\{G_n\}_{n \in \mathbb{N}_+}$, with G_3 illustrated in Fig. 8, does not have bounded decomposition width.

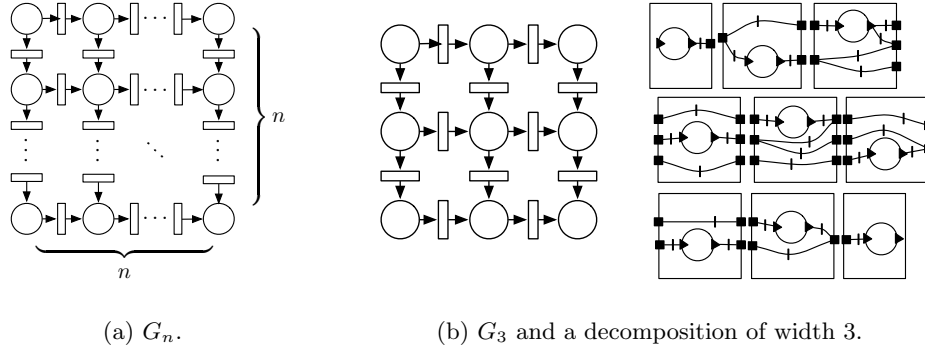


Fig. 8: Decomposing grids.

5.1 Portsets, connections and networks

For a net $N : k \rightarrow l$ and $P \subseteq \text{places}(N)$, the set of *place ports* of P is: $\text{ports}(P) \stackrel{\text{def}}{=} \{p_{\blacktriangleleft} \mid p \in P\} \cup \{p_{\blacktriangleright} \mid p \in P\}$ ⁶. When we refer to N 's *boundary ports*, we mean the elements of $[k] + [l]$. When referring to individual boundary ports we will write i_L for $(i, 0)$ and i_R for $(i, 1)$. The set of ports of N is all its place ports and boundary ports: $\text{ports}(N) \stackrel{\text{def}}{=} \text{ports}(\text{places}(N)) \cup ([k] + [l])$. We will usually refer to sets of ports as *portsets*. Given a transition t , the portset of t is:

$$\text{ports}(t) \stackrel{\text{def}}{=} \{p_{\blacktriangleright} \mid p \in {}^\circ t\} \cup \{p_{\blacktriangleleft} \mid p \in t^\circ\} \cup (\bullet t + t^\bullet).$$

We will usually write portsets using angle brackets. For instance, consider the net $R : 0 \rightarrow 1$ in Fig. 7b, with $\text{places}(R) = \{p\}$ and $\text{trans}(R) = \{t\}$. Then $\text{ports}(\{p\}) = \langle p_{\blacktriangleleft}, p_{\blacktriangleright} \rangle$, $\text{ports}(R) = \langle p_{\blacktriangleleft}, p_{\blacktriangleright}, 0_R \rangle$ and $\text{ports}(t) = \langle p_{\blacktriangleright}, 0_R \rangle$.

We will refer to sets of portsets as a *connections* and write them using square brackets. The *connection* of a port $p \in \text{ports}(N)$ is the set of portsets of all transitions that connect to p :

$$\text{conn}(p) \stackrel{\text{def}}{=} \{ \text{ports}(t) \setminus \{p\} \mid t \in \text{trans}(N) \wedge \{p\} \subset \text{ports}(t) \}.$$

For example, in $P : 1 \rightarrow 1$ in Fig. 7b: $\text{conn}(0_L) = [\langle 0_R \rangle, \langle p_{\blacktriangleleft} \rangle, \langle p_{\blacktriangleleft}, 0_R \rangle]$, $\text{conn}(p_{\blacktriangleleft}) = [\langle 0_L \rangle, \langle 0_L, 0_R \rangle]$, $\text{conn}(p_{\blacktriangleright}) = \emptyset$ and $\text{conn}(0_R) = [\langle 0_L, p_{\blacktriangleleft} \rangle, \langle 0_L \rangle]$.

We will find it useful to sometimes restrict $\text{conn}(p)$ to those sets of ports that intersect non-trivially with some subset R of the ports of a net. We write:

$$\text{conn}_R(p) = \{ K \cap R \mid K \in \text{conn}(p), K \cap R \neq \emptyset \}.$$

Suppose that $N : k \rightarrow l$ is a net with boundaries. An *oriented partition* is $\mathbf{P} = (P_l, P_r)$, where $\{P_l, P_r\}$ is a partition of $\text{places}(N)$ and $P_l, P_r \neq \emptyset$. Given an oriented partition, we define the *extended ports* of P_l and P_r : $\text{eports}(P_l) \stackrel{\text{def}}{=} \text{ports}(P_l) \cup \{(i, 0) \mid i < k\}$ and $\text{eports}(P_r) \stackrel{\text{def}}{=} \text{ports}(P_r) \cup \{(i, 1) \mid i < l\}$. These

⁶ For the sake of concreteness, $p_{\blacktriangleleft} \stackrel{\text{def}}{=} (p, \text{in})$, $p_{\blacktriangleright} \stackrel{\text{def}}{=} (p, \text{out})$.

contain the ports of the places in each set and boundary ports: P_l from the left boundary and P_r from the right boundary.

Given an oriented partition \mathbf{P} of a net N , we need to express how the places in the two disjoint place sets are interconnected. We will refer to sets of connections as *networks*. Then the network from P_l to P_r consists of the connections to extended ports of P_r , for each extended port of P_l :

$$\text{network}_{P_r}(P_l) \stackrel{\text{def}}{=} \{\text{conn}_{\text{eports}(P_r)}(p) \mid p \in \text{eports}(P_l)\}$$

and similarly for $\text{network}_{P_l}(P_r)$.

Example 19. Consider the clique $C_4 : 0 \rightarrow 0$, illustrated in Fig. 6a, and the oriented partition $\mathbf{P} = (\{0, 1\}, \{2, 3\})$. Then:

$$\begin{aligned} \text{conn}_{\{2,3\}}(0_{\blacktriangleright}) &= [\langle 2_{\blacktriangleleft} \rangle, \langle 3_{\blacktriangleleft} \rangle] = \text{conn}_{\{2,3\}}(1_{\blacktriangleright}), \\ \text{conn}_{\{2,3\}}(0_{\blacktriangleleft}) &= [\langle 2_{\blacktriangleright} \rangle, \langle 3_{\blacktriangleright} \rangle] = \text{conn}_{\{2,3\}}(1_{\blacktriangleleft}). \end{aligned}$$

Thus $\text{network}_{\{2,3\}}(\{0, 1\}) = \{[\langle 2_{\blacktriangleright} \rangle, \langle 3_{\blacktriangleright} \rangle], [\langle 2_{\blacktriangleleft} \rangle, \langle 3_{\blacktriangleleft} \rangle]\}$ and by a symmetric argument $\text{network}_{\{0,1\}}(\{2, 3\}) = \{[\langle 0_{\blacktriangleright} \rangle, \langle 1_{\blacktriangleright} \rangle], [\langle 0_{\blacktriangleleft} \rangle, \langle 1_{\blacktriangleleft} \rangle]\}$. Note that, although cliques contain many transitions, the networks between partitions are small: in fact, it is not difficult to show that for all n , any oriented partition (P_l, P_r) of the places of C_n satisfies $|\text{network}_{P_l}(P_r)| = |\text{network}_{P_r}(P_l)| = 2$. Roughly speaking, the amount of information to describe connections from one partition to another is constant, and this is the key insight that leads to the decompositions presented in Examples 17 and 18.

5.2 Bases, dimension and pure decompositions

We now show that there is a general connection between the networks of an oriented partition, and the internal boundary of any corresponding ‘;’ decompositions. First we introduce the notion of a *basis* of a network:

Definition 20 (Basis). *Given a network N , a vector of connections $b_0 \dots b_{n-1}$, is a basis of N iff $\forall c \in N$, there exists $l \subseteq [n]$ with $c = \bigcup_{i \in l} b_i$. That is, every connection in N can be written as the union of a subset of the connections of the basis. The dimension of a network N , $\dim(N)$ is the size of its smallest basis.*

Suppose we have a net $N : k \rightarrow l$ and a decomposition $N \cong N_l ; N_r$ (*) where $N_l : k \rightarrow n$, $N_r : n \rightarrow l$, with places P_l and P_r , respectively. Through slight abuse of notation we equate the places of P_l and P_r with the corresponding places in N by fixing a concrete isomorphism that witnesses (*). In particular we obtain an oriented partition (P_l, P_r) of N .

The connections of each shared-boundary port $j < n$ to N_l and N_r are just $\text{bconn}_{N_l}(j) \stackrel{\text{def}}{=} \text{conn}^{N_l}(j_R)$ and $\text{bconn}_{N_r}(j) \stackrel{\text{def}}{=} \text{conn}^{N_r}(j_L)$ where the superscripts refer to the ambient net in which the calculation takes place. We say that the composition $N_l ; N_r$ is *pure* iff, for all $j < n$ no portset in $\text{bconn}_{N_l}(j)$ contains a right boundary port i_R and, symmetrically, no portset in $\text{bconn}_{N_r}(j)$ contains a

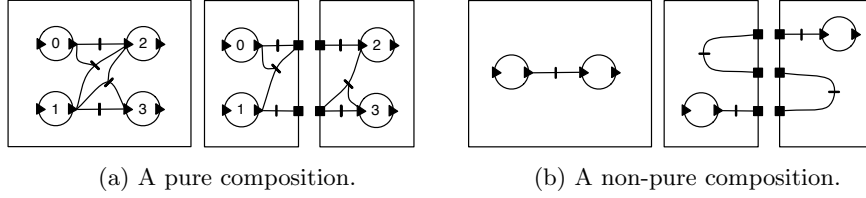


Fig. 9: Composition examples.

left boundary port i_L . In other words, no transition in N_l or N_r connects to two different shared boundary ports. It follows that in pure decompositions $\text{conn}_{N_l}(j)$ and $\text{conn}_{N_r}(j)$ are connections in N . All examples of decompositions we have considered so far are pure; a non-pure decomposition is illustrated in Fig. 9b.

Example 21. Consider the net in Fig. 9a and the corresponding pure decomposition. The shared-boundary connections are as follows:

$$\begin{aligned} \text{bconn}_{N_1}(0) &= [\langle 0_{\blacktriangleright} \rangle, \langle 0_{\blacktriangleright}, 1_{\blacktriangleright} \rangle] & \text{bconn}_{N_2}(0) &= [\langle 2_{\blacktriangleleft} \rangle] \\ \text{bconn}_{N_1}(1) &= [\langle 1_{\blacktriangleright} \rangle] & \text{bconn}_{N_2}(1) &= [\langle 2_{\blacktriangleleft}, 3_{\blacktriangleleft} \rangle, \langle 3_{\blacktriangleleft} \rangle] \end{aligned}$$

Proposition 22. *Given a net $N : k \rightarrow l$ together with a pure decomposition $N_l : k \rightarrow n, N_r : n \rightarrow l$, the vector $(\text{bconn}_{N_r}(i))_{i < n}$ is a basis for $\text{network}_{N_r}(N_l)$, and $(\text{bconn}_{N_l}(i))_{i < n}$ is a basis for $\text{network}_{N_l}(N_r)$.*

Proof. The purity of the composition implies that all transitions in the composition (minimal synchronisations) are of the form $(\{u\}, \{v\})$, $u \in \text{trans}(N_l)$, $v \in \text{trans}(N_r)$, where $u^\bullet = {}^\bullet v$, a single shared-boundary port. Then, it follows that for each $p \in \text{eports}(P_l)$:

$$\begin{aligned} \text{conn}_{\text{eports}(P_r)}(p) &= \{ \text{ports}(t) \cap \text{eports}(P_r) \mid t \in \text{trans}(N), p \in \text{ports}(t), \text{ports}(t) \cap \text{eports}(P_r) \neq \emptyset \} \\ &= \{ \text{ports}(v) \mid v \in \text{trans}(N_r), \exists u \in \text{trans}(N_l). p \in \text{ports}(u) \wedge u^\bullet = {}^\bullet v \} \\ &= \bigcup_{\{ i \mid \exists u \in N_l. p \in \text{ports}(u), p^\bullet = i \}} \text{bconn}_{N_r}(i). \end{aligned}$$

The second case follows by symmetry. □

Proposition 22 leads to the following immediate corollary.

Corollary 23. *Suppose $N : k \rightarrow l$ decomposes into $N_1; N_2$ where $N_1 : k \rightarrow n, N_2 : n \rightarrow l$. Suppose that $\mathbf{P} = (P_1, P_2)$ is the corresponding oriented partition. Then $n \geq \dim(\text{network}_{P_2}(P_1))$.* □

Example 24. Consider again the net in Fig. 9a. We have

$$\text{network}_{\{2,3\}}(\{0,1\}) = \{ [\langle 2_{\blacktriangleleft} \rangle], [\langle 3_{\blacktriangleleft} \rangle, \langle 2_{\blacktriangleleft}, 3_{\blacktriangleleft} \rangle] \}$$

It is not difficult to see that a basis of size 1 does not exist, so there is no pure decomposition into nets with places $\{0, 1\}$, $\{2, 3\}$ with size 1 boundary.

Returning to the family of grid nets G_n of Fig. 8, for any $k \in \mathbb{N}_+$, G_k has a pure decomposition of width k ; we illustrate this for G_3 in Fig. 8b, and it is not difficult to generalise the construction to arbitrary k . We omit the details here. We believe that decompositions of size $< k$ do not exist: essentially if one constructs a grid incrementally with pieces of size $< k$ one reaches a composition with boundary $> k$, using an argument similar to the statement of Corollary 23.

Example 25. Consider G_3 in Fig. 8b. We can show that there is no pure ‘;’-decomposition of width < 3 . Clearly we can assume that leaves each have fewer than 2 places. Using the conclusion of Corollary 23, we can show (by inspection) that for every “increasing” sequence of partitions of the places of G_3 , $(P_{l,1}, P_{r,1}), (P_{l,2}, P_{r,2}), \dots, (P_{l,k}, P_{r,k})$, where $|P_{l,1}|, |P_{r,k}| < 3$ and for all $1 \leq i \leq k-1$, $P_{l,i} \subseteq P_{l,i+1}$ and $|P_{l,i+1} \setminus P_{l,i}| < 3$, there exists i such that any composition $N_{l_i} : 0 \rightarrow n$, $N_{r_i} : n \rightarrow 0$ implies $n \geq 3$. We omit the tedious details. It is also not difficult to extend this argument to arbitrary pure decompositions (ie those that also have ‘ \otimes ’ nodes).

The theory of general grid partitioning is non-trivial (see, e.g. [4] for a pleasant overview) and we leave the study of this conjecture for future work.

Conjecture 26. The family $\{G_n\}_{n \in \mathbb{N}_+}$ of Fig. 8 does not have bounded decomposition width.

6 Conclusions and future work

We have considered the decomposition of 1-bounded Petri nets, employing the full algebra of nets with boundaries. Through several examples we have demonstrated that by doing so, we extend the applicability of our divide-and-conquer algorithm for reachability checking. We have introduced and examined the structural property of decomposition width for nets, and more generally, directed hypergraphs. Finally, we have developed the theory of wiring decompositions to give a lower bound on the boundary size of certain compositions.

Low decomposition width is not sufficient for avoiding state explosion when generating the transition systems from nets—this, instead, is the ‘semantic’ property referred to in the Introduction. In future work, we will consider this property, aiming to characterise the class of nets on which our technique for reachability checking is viable. Here we have concentrated on the necessary structural condition of (low) decomposition width, which also deserves further study in its own right, and how it relates to other structural properties of hypergraphs.

References

1. R. Bruni, H. C. Melgratti, and U. Montanari. A connector algebra for P/T nets interactions. In *CONCUR ‘11*, LNCS, pages 312–326. Springer, 2011.
2. R. Bruni, H. C. Melgratti, U. Montanari, and P. Sobociński. Connector algebras for C/E and P/T nets’ interactions. *LMCS*, 2012. Submitted.

3. B. Courcelle. The monadic second-order logic of graphs I. Recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.
4. W. W. Donaldson. *Grid-graph partitioning*. PhD thesis, University of Wisconsin-Madison, 2000.
5. P. Katis, N. Sabadini, and R. F. C. Walters. Representing P/T nets in Span(Graph). In *AMAST '97*, number 1349 in LNCS. Springer, 1997.
6. P. Katis, N. Sabadini, and R. F. C. Walters. Span(Graph): an algebra of transition systems. In *AMAST '97*, volume 1349 of *LNCS*. Springer, 1997.
7. K. McMillan. A technique of a state space search based on unfolding. *Form Method Syst Des*, 6(1):45–65, 1995.
8. M. Praveen and K. Lodaya. Parametrized complexity results for 1-safe Petri nets. In *CONCUR '11*, volume 6901 of *LNCS*, pages 358–372, 2011.
9. P. Sobociński. Representations of Petri net interactions. In *Concurrency Theory (CONCUR '10)*, number 6269 in LNCS, pages 554–568. Springer, 2010.
10. P. Sobociński. Nets, relations and linking diagrams. arXiv:1303.3371, 2013.
11. P. Sobociński and O. Stephens. Reachability via compositionality in Petri nets. arXiv:1303.1399v1, 2013.